Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Damjan Cvetan

Samodejno zaznavanje naprav in storitev v omrežju mobilnih naprav

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: doc. dr. Mojca Ciglarič

Ljubljana, 2014

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika dela:

Preučite tehnologije, orodja in koncepte, ki omogočajo avtomatsko konfiguracijo naprav, ki je za uporabnika videti, kot da konfiguracije sploh ni bilo (t.i. konfiguracija brez konfiguracije oziroma "zero configuration"). Osredotočite se predvsem na naslavljanje naprav, poimenovanje naprav in storitev ter odkrivanje storitev. Nato analizirajte funkcijske zahteve rešitve za prikaz navideznega reklamnega ali tehničnega gradiva obiskovalcem na mobilnih napravah npr. na sejmih ali podobnih dogodkih. Izberite primerna orodja in tehnologije za zasnovo in razvoj takšne rešitve na platform iOS, pozorni bodite tako na pogled skrbnika kot tudi na pogled uporabnika. Rešitev preizkusite in pokažite, da zadošča zahtevam, komentirajte pa tudi njen varnostni nivo.

Mentor: doc. dr. Mojca Ciglarič

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Damjan Cvetan, z vpisno številko **63060046**, sem avtor diplomskega dela z naslovom:

Samodejno zaznavanje naprav in storitev v omrežju mobilnih naprav

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Mojce Ciglarič,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 15. aprila 2014 Podpis avtorja:

Diplomsko delo je rezultat enačbe mnogih spremenljivk, ki so različno vplivale na potek študija in osebni razvoj. Kljub malo daljšemu študijskemu obdobju, se je enačba izenačila in ob tem se iz srca zahvaljujem svoji mentorici, doc. dr. Mojci Ciglarič, za pomoč in svetovanje pri izdelavi diplomskega dela. Posebej se zahvaljujem svoji družini, punci Maji in prijateljem, ki so me podpirali in stali ob strani vsa leta študija. Hvala!

Delo posvečam svoji družini, ki mi je ves čas mojega izobraževanja stala ob strani. Nudili so mi podporo in mi omogočili študij, ki sem ga izbral.

Kazalo

Seznam kratic

API	Application Programming Interface
ARP	Address Resolution Protocol
D-Bus	Sistemsko sporočilno vodilo za komunikacijo med aplikacijami
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DNS-SD	Domain Name System Service Discovery
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IDE	Integrated development environment
IETF	Internet Engineering Task Forse
iOS	iPhone OS
IP	Internet Protocol
IPP	Internet Printing Protocol
IPv4	Internet Protocol version Four

IPv4LL Internet Protocol version Four Link-Local addressing

IPv6 Internet Protocol version Six

LAN Local Area Netwrok

LDAP Lightweight Directory Access Protocol

LGPL Lesser General Public License

LPR Line Printer Remote protocol

MAC Media Access Control Address

mDNS Multicast Domain Name System

PTR Domain Name Pointer

QR Code Quick Response Code

RFC Request For Comments

RR Resource Record

SRV Domian Name Service Record

UNIX Večopravilni, večuporabniški operacijski sistem, ki obstaja v različnih izdajah.

US-ASCII US - American Standard Code for Information Interchange

vFlyer Virtual Flyer iOS aplikacija

VPN Virtual Private Network

XML Extensible Markup Language

Povzetek

Povezovanje dveh naprav naj bi bilo vedno bolj enostavno in nekaj, kar zmore vsak povprečni uporabnik informacijskih storitev. Za doseganje takšnih želja je potrebno poskrbeti za samodejno nastavljanje vseh naprav vključenih v povezavo. V diplomskem delu sem preučil načine samodejnega nastavljanja dveh naprav, med katerima želimo vzpostaviti povezavo. Osnovni pogoj za začetek nastavljanja je, da imajo vse naprave veljaven naslov IP, preko katerega se izvaja vsa nadaljnja komunikacija. Naslavljanje se lahko izvaja na več načinov, odvisno od omrežja v katerem se naprava nahaja. Te načine sem si podrobneje ogledal, preizkusil in opisal.

Diplomsko delo v nadaljevanju predstavlja tako načine oglaševanja storitve v omrežju, kot tudi načine odkrivanja le teh. Za prikaz praktičnega delovanja sem razvil aplikacijo na platformi iOS. Ta omogoča enostavno povezovanje do naprav delujočih v kiosk načinu z eno aplikacijo, katere želimo upravljati na daljavo. Preučil sem načine implementacije konfiguracije brez konfiguracije za platformo iOS in izbral najbolj ustrezno.

Ključne besede

Brez konfiguracije, Bonjour, mDNS, način enojne aplikacije, iOS

Abstract

Connecting two devices is supposed to be a simple task that an average user can do. For achieving such simplicity of use, it is necessary to provide automatic setup of all participating devices. I have examined different ways of automatic setup between two interconnected devices. The fundamentals of every connection setup are valid IP addresses on both devices which are used for all further communications. Depending on the network, addressing can be done in several ways. I took a closer look at all of them, tested them and described them in details.

In the thesis I describe how service broadcast and service discovery can be done. I have studied different ways of Zero-Configuration implementation on iOS platform, chose the most suitable one and included it in my aplication. I have also developed an example application for iOS devices. Application offers extremely easy way of communicating among devices that are running single application in kiosk mode. It enables easy management of remote devices' content.

Key words

Zero-configuration, Bonjour, mDNS, Single APP mode, iOS

Poglavje 1

Uvod

Mobilne naprave so si že krepko utrle pot v naš vsakdan in ne uporabljamo jih samo v prostem, temveč vedno bolj tudi v delovnem času. Uporabnost mobilnih naprav raste s številom mobilnih aplikacij in storitev, ki nam pohitrijo vsakdanja opravila, oziroma nam omogočajo opravljanje nekaterih opravil na poti. Opaziti je tudi porast namenskih aplikacij, ki jih zaposleni uporabljajo znotraj svojega podjetja. Takšnih primerov je mnogo, nekaj izmed njih uporabnikom omogoča upravljanje naprav, izdelovanje miselnih vzorcev in načrtov, statistične obdelave in analize, glasovne in video pogovore.

Marsikatero podjetje, ki želi svojim strankam na sejmih ali drugih lokacijah predstaviti produkte, tiska vsemogoče vrste oglasnega gradiva, za katerega se navadno porabi vedno več sredstev. Kot alternativo tiskanim gradivom, bi lahko za prikaz reklamnega ali tehničnega gradiva uporabljali tablice ali mobilne telefone in tako omilili stroške. Omenjene naprave je mogoče uporabljati samostojno, ali pa kot vir video povezave za samodejno predvajanje na večjih zaslonih.

V primeru uporabe takšnega pristopa naletimo na ovire, ki jih moramo premagati:

- Obiskovalec ima v času uporabe naprave nadzor nad njo in ne želimo, da bi se z uporabo drugih aplikacij in možnosti oddaljil od našega produkta.

- Na množično obiskanih dogodkih, kjer imamo brezžičen dostop do spleta, se velikokrat srečamo s težavo, da povezava do spleta ne deluje ali pa je kako drugače omejena in je zato ogled vsebin preko internetnega omrežja otežen.

- Naprave za prikaz vsebin na večjih zaslonih so lahko na težko dostopnem mestu, kar nam oteži posodobitev vsebine.

Da bi se izognili omenjenim težavam, bomo za omejitev uporabnikovega dostopa na napravi uporabili kiosk programsko opremo ter način shranjevanja gradiva na napravo, kar nam omogoča prikaz gradiva v načinu brez omrežne povezave. Kiosk programsko opremo sestavljata za to posebej razviti sistem in uporabniški vmesnik. Naloga takšne programske opreme je, da zaklene uporabnika na samo eno izbrano aplikacijo. Uporabniku so tako na voljo samo opravila znotraj zaklenjene aplikacije, pri čemer niti nima dostopa do operacijskega sistema. [5]

Kljub pozitivnim lastnostim kiosk programske opreme, pa se pojavi vprašanje, kako omogočiti dostop administratorju, ki navadno potrebuje večji nivo dostopa kot pa končni uporabniki. Navadno se za takšne namene omogoči oddaljen nadzor in upravljanje aplikacije, kar je tudi način, ki ga bomo uporabili v naši rešitvi. Zaradi že omenjene omejene internetne povezljivosti, mora oddaljeni nadzor delovati znotraj lokalnega omrežja in administratorju omogočiti enostavno povezovanje.

Z namensko aplikacijo Virtual Flyer iOS aplikacija (vFlyer) bomo tako izvedli enostavno pregledovanje vsebine za uporabnike, kot tudi enostavno upravljanje vsebine in nadzor nad napravami za administratorje. Aplikacija bo lahko delovala na dva različna načina:

1. **Predstavitveni način** je namenjen uporabniku za enostavno pregledovanje gradiva. Gradivo bo predstavljeno na straneh med katerimi je možno prehajati z uporabo gest zaslona.

2. **Krmilni način** je namenjen krmiljenju ostalih naprav. Administratorju takšen način omogoča povezovanje na naprave, ki delujejo v pred-

stavitvenem načinu. S takšnim dostopom bo lahko dodal novo gradivo, izbrisal obstoječe gradivo in spremenil vrstni red gradiva.

Pri obeh načinih delovanja je zelo pomembno kako se krmilna naprava povezuje s predstavitveno. Navadili smo se, da moramo pri povezovanju dveh ali več naprav programu podati informacijo o načinu povezave, naslovu oddaljene naprave, potrebnemu protokolu in morda tudi številko vrat. V našem primeru se bodo naprave povezovale preko Internet Protocol (IP) komunikacije, pri čemer je potrebno poznati vsaj naslov IP ponora in številko vrat. Iskanje naslova IP in vpisovanje tega v za to namenjeno vnosno polje, lahko neizkušenemu uporabniku vzame veliko nepotrebnega časa. V naši rešitvi se bomo takšnim zapletenim postopkom skušali izogniti.

Samodejno odkrivanje in povezovanje naprav nam namreč omogoča enostavno povezovanje naprav. Slednje bomo poleg ostalega implementirati v aplikaciji, ki bo delovala na platformi iOS. [4, 18]

Cilj diplomskega dela je preučiti tehnologije, ki omogočajo samodejno odkrivanje naprav in njihovih storitev v omrežju, ter povezovanje teh storitev brez poznavanja naslova IP in številke vrat potrebnih za povezavo. Preučiti je potrebno načine naslavljanja naprav, njihovo odkrivanje in poizvedovanje po vrsti storitve, ki nas zanima. Izmed ustreznih tehnologij je potrebno izbrati tisto, ki bo kar se da najbolje reševala to težavo na platformi iOS in jo uporabiti pri razvoju rešitve oddaljenega upravljanja naprav v kiosk načinu. Cilj diplome bo dosežen, ko bo prototipna rešitev:

- Na napravi v kiosk načinu, v omrežju uspešno oglaševala svojo prisotnost, svoje ime, vrsto storitve in nenazadnje naslov IP in številko vrat.
- V nadzornem načinu na omrežju samodejno zaznala prihod in odhod aplikacij v predstavitvenem načinu in skladno s tem osveževala seznam.
- Uspešno vzpostavila komunikacijo med aplikacijama v dveh različnih načinih in uspešno upravljala oddaljeno napravo.

Poglavje 2

Namestitev omrežnih naprav

2.1 Naslavljanje

Naprave v računalniškem omrežju, ki uporabljajo IP za komunikacijo potrebujejo naslov IP, katerega lahko pridobijo na več načinov. Naslov lahko napravi dodelimo sami, ali imamo v omrežju Dynamic Host Configuration Protocol (DHCP) strežnik, ki dodeljuje naslove glede na njegove nastavitve, ali pa naprava samodejno določi link-local naslov. Ne glede na način dodelitve IP naslova moramo za vzpostavitev povezave poznati naslov naprave, na katero se želimo povezati, vrsto storitve, ki jo bomo uporabljali, in številko vrat preko katerih bomo vzpostavili povezavo. Naša naloga je poskrbeti, da uporabniku ni potrebno poznati teh podatkov in mu tako le ponuditi storitve, ki so v omrežju na voljo.

2.2 Poimenovanje

Naslove IP lahko predstavimo v tekstovni obliki, pri čemer potrebujemo Domain Name System (DNS) strežnik, ki dela preslikavo med tekstovnim imenom naprave in logičnim naslovom. Tako se na primer ime *localhost* preslika v 127.0.0.1 ali pa ime *www.uni-lj.si* v 91.216.54.5. DNS strežnik lahko beleži povratne Domain Name Pointer (PTR) zapise ter nam tako pove ime naprave

RR	Namen
A	IPv4 naslov naprave
AAAA	IPv6 naslov naprave
CNAME	Kanonično ime za psevdonim
PTR	Domain name pointer
TXT	Text strings
SRV	Location of services

Tabela 2.1: Opis nekaterih RR zapisov

za iskani naslov IP. [13]

Strežniki DNS lahko vsebujejo več informacij kot le ime računalnika, domeno in naslov IP. Vsebujejo tudi tako imenovane Resource Records (RRs) zapise, ki so bili določeni v več Request For Commentss (RFCs) dokumentih objavljenih s strani Internet Engineering Task Forse (IETF).

2.2.1 SRV

Domian Name Service Record (SRV) RR nam omogoči uporabo večjega števila strežnikov za eno samo domeno, kjer lahko odjamalec povpraša po določeni storitvi ali protokolu in kot odgovor dobi imena razpoložljivih strežnikov. Z uporabo SRV mehanizma lahko naredimo DNS poizvedbo po SRV zapisu *_http._tcp.primer.si*. Pri tem sprašujemo po Hypertext Transfer Protocol (HTTP) storitvi za domeno *primer.si*. [6]

Oblika SRV RR zapisa na strani DNS strežnika:

```
_Service._Proto.Name TTL Class SRV Priority Weight Port Target
```

- **_Service**. Simbolično ime storitve. Podčrtaj na začetku (_) je dodan v izogib trkov z DNS zapisi.
- **_Proto**. Ime zahtevanega protokola. Najbolj uporabni vrednosti sta *_TCP* in *_UDP*.

- **Name**. Ime domene na katero se ta zapis navezuje.
- **TTL**. Časovni interval po katerem se predpomnjen zapis osveži. [13]
- **Class**. Dvoznakovna okrajšava DNS CLASS vrednosti. [13]
- **Priority**. Prioriteta ciljnega strežnika. Odjemalec mora skušati vzpostaviti povezavo s strežnikom najnižje vrednosti. Strežniki z isto vrednostjo se nadalje razvrščajo glede na *Weight* vrednost.
- **Weight**. Mehanizem za izbiro strežnika.
- **Port**. Številka vrat storitve v razponu od 0 do 65535.
- **Target**. Ime domene ciljnega strežnika. Obstajati mora eno ali več imen in ne smejo biti psevdonimi.

_Service lahko vsebuje največ 15 znakov kodne tabele US - American Standard Code for Information Interchange (US-ASCII), vsebuje lahko črke, številke in vezaje. Za register tipov storitev in potrebnih vrat za povezovanje skrbi Internet Assigned Numbers Authority (IANA), ki na svojo spletni strani ponuja seznam vseh dodeljenih tipov storitev. Poleg tega je možno na naslovu `http://www.iana.org/form/ports-services` oddati zahtevo za registracijo storitve. [3, 12]

2.2.2 TXT

TXT RR zapis je splošen zapis, ki dovoljuje uporabo navadno berljivega besedila znotraj DNS zapisov domene. Uporabljamo jih za zapis dodatnih informacij, ki bi jih morda odjamalec potreboval. Kot primer naj opišem Line Printer Remote protocol (LPR) protokol, ki se uporablja na Večopravilni, večuporabniški operacijski sistem, ki obstaja v različnih izdajah. (UNIX) sistemih za tiskanje. LPR ne določa nikakršnega mehanizma za poizvedovanje parametrov tiskalnika. Namesto da bi tiskalnik oglaševal LPR zmožnosti, so le te vgrajane v TXT zapis. Pri tem se za zapis uporabi kombinacija parov *ključ/vrednost*.

2.2.3 PTR

PTR zapis preslika logični naslov v kanonično ime (vzdevek) za tega gostitelja. Ta zapis se obravnava kot enostaven podatek, saj ne potrebuje nobene nadaljnje obdelave kot jo potrebujemo naprimer pri CNAME zapisih.

Koda 2.1: PTR poizvedba

```
1 $ host 193.2.1.87
2 87.1.2.193.in-addr.arpa domain name pointer kanin.arnes.si.
```

Poglavje 3

Konfiguracija brez konfiguracije

Konfiguracija brez konfiguracije je arhitektura oziroma kombinacija treh tehnologij, ki skupaj igrajo pomembno vlogo uporabniške izkušnje. Skupaj te tehnologije delujejo na način, da uporabniku za povezovanje naprav oziroma storitev na napravah ni potrebno poznati podrobnih informacij, kot so naslov IP ali gostiteljevo ime, številko vrat in vrste protokola. [15, 18]

Konfiguracija brez konfiguracije skrbi za tri pomembne zadeve, ki so potrebne za doseganje cilja. To so:

- Avtomatično naslavljanje.
- Translacija med imenom in naslovom ter obratno.
- Odkrivanje storitev.

3.1 Naslavljanje

3.1.1 Ročno naslavljanje

Ročno naslavljanje pomeni, da za vsako napravo v omrežju ročno vnesemo naslov IP, masko omrežja, prehod, strežnike DNS in druge informacije, ki jih naše omrežje zahteva. Pri takšnem naslavljanju moramo sami skrbeti, da

ima vsaka naprava unikaten naslov, saj lahko v primeru podvajanja pride do trkov in nedelovanja.

Koda 3.1: Ročna dodelitev naslova IP

```
1 auto eth0
2 iface eth0 inet static
3   address 192.168.1.10
4   netmask 255.255.255.0
5   network 192.168.1.0
6   broadcast 192.168.1.255
7   gateway 192.168.1.1
8   dns-nameservers 8.8.8.8
```

3.1.2 Samodejno naslavljanje z DHCP

DHCP strežnik, kot lahko razberemo iz imena, skrbi za dinamično dodeljevanje naslovov IP. V splošnem deluje tako, da novi napravi v omrežju dodeli prosti naslov iz množice naslovov, ki so mu na voljo. Pri tem ostane naslov rezerviran za to napravo določen čas in v primeru, da se naprava znotraj tega časa ponovno pojavi v omrežju, le ta pridobi isti naslov. V nasprotnem primeru DHCP strežnik izbere novi naslov, saj se lahko zgodi, da je pretekli naslov že v uporabi.

Kljub dinamičnemu naslavljanju pa lahko z ustreznimi nastavitvami poskrbimo, da naprave v omrežju pridobijo vedno isti naslov. To storimo tako, da v nastavitvah strežnika zapišemo pare naslovov IP in Media Access Control Address (MAC).

Koda 3.2: DHCP nastavitve

```
1 subnet 192.168.1.0 netmask 255.255.255.0 {
2   option domain-name-servers 8.8.8.8, 8.8.4.4;
3   option subnet-mask 255.255.255.0;
4   option broadcast-address 192.168.1.255;
```

```
 5   option routers 192.168.1.1;
 6   ddns-updates off;
 7   default-lease-time 600;
 8   max-lease-time 600;
 9
10   pool {
11     range 192.168.1.10 192.168.1.150;
12     allow unknown-clients;
13     allow known clients;
14   }
15 }
```

3.1.3 Samooklicano naslavljanje

V zgornjih primerih smo obravnavali naslov IP iz omrežja *192.168.1.0/24*. Slednje nam je verjetno dobro poznano, saj ga srečamo kot prevzeto nastavitev v večini Local Area Netwrok (LAN) omrežij. To omrežje je del omrežja, ki je s strani IANA določen in rezerviran za privatno uporabo. Podobno obstaja več različnih omrežij, ki so rezervirana za različne namene: [7]

Omrežje	Namen uporabe
10.0.0.0/8	Privatna omrežja
172.16.0.0/12	Privatna omrežja
192.168.0.0/16	Privatna omrežja
127.0.0.0/8	Povratna zanka
169.254.0.0/16	Link Local

Tabela 3.1: Seznam nekaterih rezerviranih omrežij.

V omrežjih kjer nimamo ročnega naslavljanja in smo odvisni od DHCP strežnika ali pa v omrežjih, kjer slednjega sploh nimamo, je vseeno pomembno, da obstajajo načini na katere naprava pridobi naslov vsaj za lokalno komunikacijo. Tu naletimo na link-local (Internet Protocol version

Four Link-Local addressing (IPv4LL)) omrežje, ki ga navajamo kot zadnjega v tabeli 3.1 in je primarno uporabno za omrežja 2, 10 ali celo 100 naprav. Analize takšnih omrežij so celo pokazale, da deluje dobro tudi v primeru več kot 1000 naprav. Tudi Internet Protocol version Six (IPv6) pozna link-local naslavljanje in je ne samo enostavnejše od Internet Protocol version Four (IPv4) link-local naslavljanja, temveč tudi bolj zanesljivo. IPv6 link-local naslov je iz omrežja *fe80::/10* in je dodeljen z stateless ali stateful mehanizmi. [8, 14]

Izbira naslova

Izbira naslova poteka tako, da gostitelj z uporabo psevdonaključnega generatorja [1] izbere naslov IP iz območja *169.254.1.0 - 169.254.254.255*. Opazimo lahko, da to območje ne predstavlja omrežja *169.254.0.0/16* v celoti, saj je prvih in zadnjih 256 naslovov tega omrežja rezerviranih za uporabo v prihodnosti in ne smejo biti izbrani s strani gostitelja. Gostitelj naslov izbere s pomočjo psevdonaključnega generatorja, s čimer različni gostitelji ustvarijo različno zaporedje števil. Generatorju podamo začetno vrednost (seme), s katerim omogočimo poljubno začetno stanje generatorja. Primerna izbira začetne vrednosti je naslov MAC, ki se med napravami razlikuje. S tem dobimo na določeni napravi vedno isti naslov, kar je veliko boljše, kot pa če bi za seme uporabili uro, kjer bi dobili vedno drugačne naslove in v primeru zagona naprav ob istem času tudi iste naslove v omrežju.

Potrditev naslova

Gostitelj nima pravice uporabljati naslova, dokler ni prepričan, da nobena druga naprava v omrežju ne uporablja tega naslova. Ko naprava izbere naslov, le ta v omrežje pošlje nekaj Address Resolution Protocol (ARP) zahtev, kjer se zanima za naslov MAC v povezavi z izbranim naslovom IP.

[1] `http://en.wikipedia.org/wiki/Pseudorandom_number_generator`

Koda 3.3: ARP promet

```
1 $ tcpdump -i eth0 arp
2 tcpdump: verbose output suppressed, use -v or -vv for full
     protocol decode
3 listening on eth0, link-type EN10MB (Ethernet), capture size
     96 bytes
4 19:41:52.809642 arp who-has 169.254.1.86 tell 169.254.1.86
5 19:41:52.863689 arp who-has 169.254.1.245 tell 169.254.1.245
6 19:41:53.024769 arp who-has 169.254.1.10 tell 0.0.0.0
```

Zadnja vrstica s koncem *tell 0.0.0.0* napravi pove, da naslov *169.254.1.10* ne uporablja nobena druga naprava v omrežju.

Naznanjanje naslova

Ko naprava izbere prosti naslov, je pomembno, da o tem obvesti druge naprave v omrežju. Lahko se namreč zgodi, da ravno v času med potrditvijo prostega naslova do izbire v omrežje pride naprava s tem naslovom. Lahko se zgodi tudi, da imajo druge naprave v omrežju ARP zapise lokalno pomnene in jih je potrebno osvežiti. Z naznanjanjem naslova poskrbimo, da se staro stanje ARP tabel posodobi tako, da kažejo na naš MAC naslov.

3.2 Poimenovanje

Za poimenovanje naprav lahko uporabljamo strežnik DNS, o katerem smo že govorili. Slednji je lahko nameščen lokalno ali pa zunaj LAN omrežja in če katerikoli gostitelj želi pretvoriti ime gostitelja v naslov, to stori s poizvedbo na strežnik DNS.

To pa ni vedno dovolj, saj se nam podobno kot pri naslavljanju zgodi, da omrežje strežnika DNS nima ali pa je le ta nedelujoč. Podobno funkcionalnost lahko v takšnih primerih dosežemo z Multicast Domain Name System (mDNS). [16]

mDNS deluje podobno kot normalni strežnik DNS, vendar namesto da bi gostitelj poizvedbo poslal strežniku DNS, jo pošlje kot multicast večji skupini gostiteljev, kjer odgovor poda gostitelj z iskanim imenom. Poglavitne prednosti mDNS imen so, da zahtevajo malo ali nič administracije ali namestitve, da delujejo tudi, ko strežnikov DNS v omrežju ni na voljo ter delujejo tudi ob napakah v omrežju.

Ime gostitelja in storitve

Ime gostitelja in storitve je sestavljeno iz:

```
Service Instance Name = <Instance> . <Service> . <Domain>
```

<Instance> del imena je uporabniku prijazno ime samovoljno sestavljeno iz znakov Net-Unicode kodne tabele. To ime naj bi bilo nastavljivo s strani uporabnika in ne sme biti določeno s strani proizvajalca strojne ali programske opreme. Pri imenu niti ni potrebno skrbeti, da je unikatno, kot bi morda mislili. Če pride do podvajanja imena v času oglaševanja storitve s strani Domain Name System Service Discovery (DNS-SD) preko mDNS, pride do samodejne odprave konflikta imen. Pri zaznavi konflikta mora storitev: [16, 17]

1. Avtomatično izbrati novo ime. Navadno tako, da se obstoječemu imenu pripne zaporedno številko.
2. Skušati oglaševati storitev z novim imenom.
3. V primeru uspeha trajno shraniti ime.

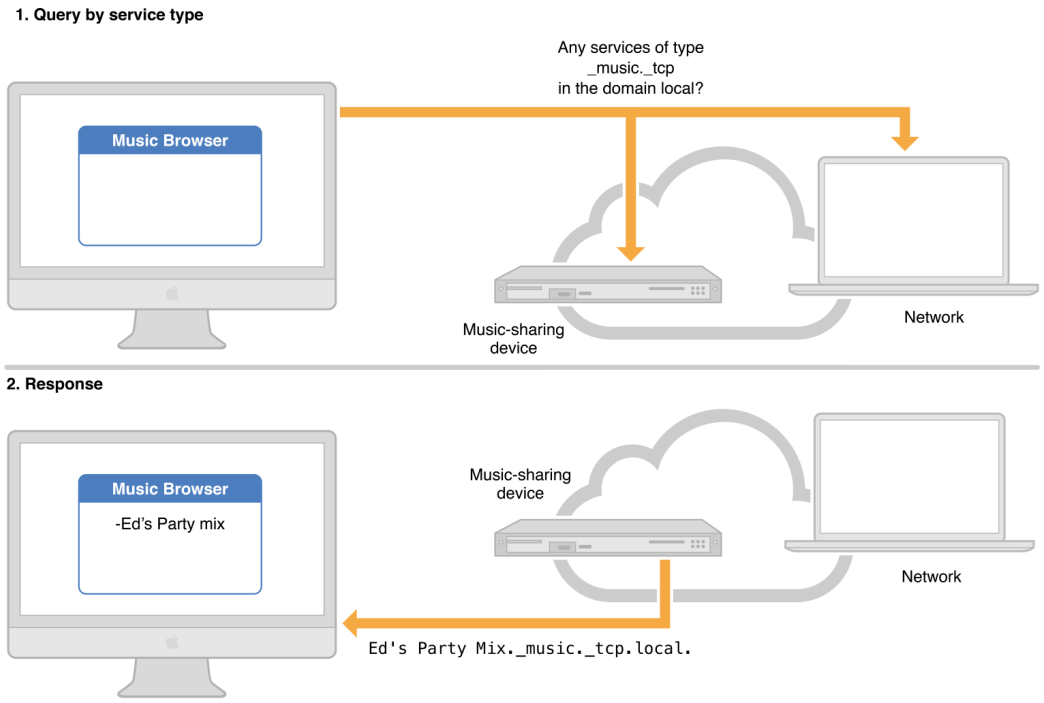
3.3 Odkrivanje storitev

S tradicionalnim DNS SRV zapisom lahko za izbrano domeno pridobimo seznam strežnikov, ki ponujajo izbrano storitev. Tako naprimer SRV zapis *_http._tcp.primer.si* odjemalcem omogoči, da najdejo strežnike, ki omogočajo

_http._tcp storitve (spletno stran) za domeno *primer.si*. Z vidika odjemalca vsi strežniki ponujajo isto spletno stran in za njega ni pomembno, kateri strežnik bo izbral, le da ga izbere glede na pravila *weight* in *priority* vrednosti zapisa SRV (Poglavje 2.2.1).

Takšen pristop je za našo uporabo neprimeren, saj bi v primeru poizvedbe po *_ipp._tcp.podjetje.si* res dobili seznam tiskalnikov Internet Printing Protocol (IPP), vendar bi bila izbira naključnega tiskalnika uporabniku zelo neprijazna.

Mehanizem DNS-SD si tako od SRV zapisov sposodi le poimenovanje storitev in pošlje poizvedbo za PTR zapis, ki je kazalec od enega imena do drugega. Odkrivanje storitev tako poteka s pomočjo DNS zapisov s poizvedbo za PTR zapis, ki se ujema z vrsto storitve kot na primer *_ipp._tcp.podjetje.si*. Odgovor v PTR obliki poda strežnik mDNS gostitelja, ki storitev ponuja tako, da vsebuje ime gostitelja.



Slika 3.1: Odkrivanje storitev za deljenje glasbe. Vir: iOS Developer Library, Bonjour Operations. [11]

Na sliki 3.1 vidimo, da odjemalec išče storitve za deljeno glasbo. V prvem koraku aplikacija izvede poizvedbo *_music._tcp* v *.local* domeni na standardni multicast naslov *224.0.0.251*. Vsak mDNS strežnik v omrežju dobi to zahtevo, vendar samo naprave, ki ponujajo iskano storitev, odgovorijo z PTR zapisom (drugi korak). PTR zapis *Ed's Party Mix._music._tcp.local* v našem primeru vsebuje ime ponudnika storitve *Ed's Party Mix*, ki ga aplikacija lahko doda na seznam na zaslonu in prikaže uporabniku.

3.4 Razreševanje

Odkrivanje storitve (poglavje 3.3) se navadno ne izvaja pogosto, oziroma se izvede, ko na primer uporabnik izbere tiskalnik za tiskanje. Izbira tiskalnik shrani celotno vrednost zapisa PTR, medtem ko številko vrat in naslova IP

ne shranjuje. Takšna ločitev je posledica dejstva, da se lahko ti podatki spreminjajo iz dneva v dan, kar pa ne velja za ime gostitelja.

Zaradi tega se razreševanje storitve v naslov IP in številko vrat izvaja šele, ko uporabnik želi storitev uporabljati. V ta namen aplikacija izvede SRV poizvedbo z imenom storitve.

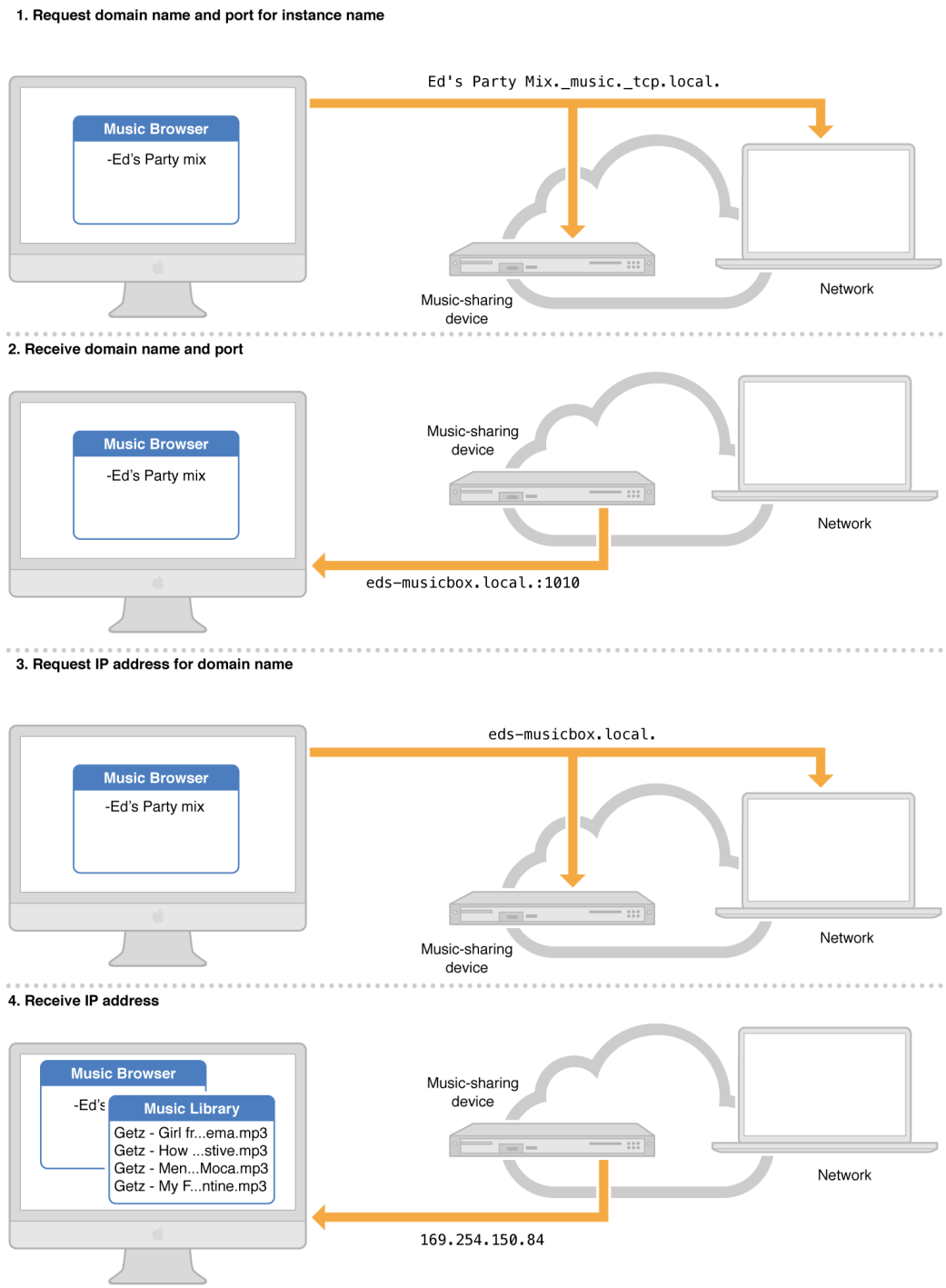
Slika 3.2 prikazuje potrebne korake v nalogi razreševanja. Cilj naloge je, da iz shranjenega PTR zapisa pridobimo potrebne podatke za vzpostavitev povezave. To so naslov IP in številka vrat.

1. Začetek razreševanja storitve se začne z DNS poizvedbo na multicast naslov *224.0.0.251*, ki sprašuje po *Ed's party Mix._music._tcp.local* SRV vrednosti.

2. mDNS odgovori na poizvedbo z imenom gostitelja *eds-musicbox.local.* in številko vrat *1010*.

3. mDNS pošlje poizvedbo po naslovu IP (A/AAAA DNS zapis) za ime gostitelja *eds-musicbox.local*.

4. mDNS odgovori z ustreznim naslovom IP, ki je v našem primeru *169.254.150.84*.



Slika 3.2: Razreševanje naslova storitve. Vir: iOS Developer Library, Bonjour Operations. [11]

3.5 Implementacije arhitekture brez konfiguracije

3.5.1 Avahi

Slika 3.3: Avahi logotip.

Avahi je prosto kodna programska implementacija konfiguracije brez konfiguracije za Linux in BSD operacijske sisteme in je že vključena v večino Linux distribucij ter vsebuje implementacije omenjenih tehnologij IPv4LL, mDNS in DNS-SD. Avahi projekt se je začel zaradi Applove implementacije Bonjour, ki je bila izdana pod GPL Apple Public Source licenco. Še predno je bil Bonjour prelecinciran pod Apache licenco, je Avahi že postal standardna implementacija v brezplačnih operacijskih sistemih GNU/Linux. [1, 2, 19]

Za komunikacijo med uporabniškim vmesnikom in Avahi demonom se uporablja Sistemsko sporočilno vodilo za komunikacijo med aplikacijami (D-Bus). Demon je odgovoren za usklajevanje in medpomnjenje poizvedb in odgovorov z namenom zmanjševanja prometa, ki se pri tem ustvarja v omrežju.

Avahi je izdan pod Lesser General Public License (LGPL) licenco ter vsebuje nekaj izrazitih funkcionalnosti kot so:

- Podpora za IPv4 in IPv6.
- Opusti vse privilegije in se izvaja kot "avahi"uporabnik.
- Podpora za chroot().
- Podpora za nalaganje statičnih nastavitev z Extensible Markup Language (XML) dokumentom.

- Možnost odražanja mDNS prometa med več podomrežji.
- Združljivost knjižnjice z ostalima izvedbama konfiguraicje brez konfiguracije HOWL in Bojour.

Koda 3.4: Avahi XML za storitev File Transfer Protocol (FTP)

```
1 <?xml version="1.0" standalone='no'?>
2 <!DOCTYPE service-group SYSTEM "avahi-service.dtd">
3 <service-group>
4   <name>FTP file sharing</name>
5   <service>
6     <type>_ftp._tcp</type>
7     <port>21</port>
8   </service>
9 </service-group>
```

3.5.2 Avahi in Bonjour

Zmogljivosti Avahi arhitekture so zelo podobne Bonjour-ovim, oziroma naj bi bile v določenih primerih celo boljše. Kljub temu pa je znano, da Avahi lahko izgubi zmožnost delovanja v primeru upravljanja več poizvedb hkrati.

3.5.3 Apple Bonjour

Bonjour je implementacija konfiguracije brez konfiguracije s strani Apple Inc. Podpira oglaševanje in odkrivanje storitev na učinkovit in robusten način z uporabo mDNS in link-local naslavljanja, ko je to potrebno.

Link-local naslavljanje, imensko razreševanje in odkrivanje storitev lahko v omrežju drastično povečajo promet, vendar Bonjour izvaja več korakov, da zmanjša ta promet kolikor je le mogoče. V ta namen izvaja medpomnjenje mDNS zapisov, zatiranje podvojenih odgovorov, Exponential Back-off in oglaševanje storitev. [8]

Slika 3.4: Bonjour logotip.

Medpomnjenje

Bonjour uporablja medpomnjenje mDNS zapisov, da prepreči gostiteljem ponovno zahtevanje informacij, katere so že enkrat zahtevali. Ko na primer gostitelj zahteva seznam LPR čakalnih vrst, dobi multicast odgovor tako, da ga vidijo vsi lokalni gostitelji. Ko eden izmed gostiteljev nato potrebuje isti seznam, je le ta že medpomnjen in tako ne rabi ponovno poslati poizvedbe.

Zatiranje podvojenih odgovorov

Za preprečevanje ponavljajočih se odgovorov za isto poizvedbo Bonjour hrani seznam poznanih odgovorov. Če gostitelj na primer išče tiskalnike v omrežju in je seznam poznanih odgovorov prazen, dobi recimo odgovor od šestih (6) razpoložljivih tiskalnikov. Naslednjič ko gostitelj poizveduje po tiskalnikih, poizvedba že vsebuje seznam poznanih strežnikov. Tiskalniški strežniki, ki so že na seznamu, zaradi tega ne bodo podali odgovora.

Exponential Back-off in oglaševanje storitev

Namesto da bi gostitelj nenehno pošiljal poizvedbe po stortivah, počne to tako, da pošlje eno začetno poizvedbo in nato vse nadaljnje manj pogosto. Na primer po 1., 3., 9., 27. sekundah in tako naprej, do največ ene (1) ure. To pa ne pomeni, da potrebujemo eno uro, da najdemo nove storitve. Ko je nova storitev v omrežju, le ta nekajkrat oznani svojo prisotnost s podobnim Exponential Back-off algoritmom.

Poglavje 4

Izvedba rešitve Virtual Flyer

vFlyer je aplikacija za naprave z iPhone OS (iOS) operacijskim sistemom, ki smo jo razvili v sklopu naloge za prikaz praktičnega delovanja konfiguracije brez konfiguracije.

Aplikacija deluje v dveh načinih, med katerima je eden delovanje aplikacije v kiosk načinu, kjer nimamo možnosti posodabljanje vsebine. Za posodobitev vsebine potrebujemo ločeno napravo, ki mora biti v istem omrežju kot prvotna naprava. Aplikacija za povezovanje med napravama ne potrebuje nikakršnega vnosa naslova IP, vrat ali česarkoli drugega.

4.1 Kiosk način

Kiosk način na napravah z iOS verzijo 7.0 ali novejšo lahko uveljavimo tako da uporabnika zaklenemo v uporabo ene same aplikacije preko konfiguracijskega profila. Konfiguracijski profil je datoteka XML, ki nam omogoča porazdeliti različne nastavitve na eno ali več naprav. Uporaba profilov je močno dobrodošla v okoljih z večjim številom naprav in nastavitvami, ki se na večini naprav ponavljajo. Nastavitve lahko zajemajo podatke za nastavitev brezžičnih povezav, Virtual Private Network (VPN) storitev, e-poštih predalov, Lightweight Directory Access Protocol (LDAP) storitev in drugih. XML datoteka vsebuje zapise formata *.plist* (Property list). [9]

Za uspešno delovanje kiosk načina je potrebno v XML vsebino dodati nastavitve za poseben *PayloadType* z vrednostjo *com.apple.app.lock* in z vsemi nujno potrebnimi atributi in vrednostmi. Na eni napravi lahko obstaja samo eden takšen tip vsebine, ki ga je mogoče namestiti le na *nadzorovane* naprave.

Pri namestitvi vsebine za *com.apple.app.lock* se naprava zaklene na eno aplikacijo, ki jo definiramo v nastavitvah. S tem se onemogoči gumb *domov* ter določi aplikacijo, katero bo naprava odprla v primeru vrnitve iz spanja ali ponovnega zagona.

Ključ	Tip	Vrednost
App	Slovar	Slovar, ki vsebuje informacije o aplikaciji. Možni ključi slovarja so *Identifier*, *Options*, *UserEnabledOptions*.
Identifier	Niz	Niz znakov, ki enoznačno določajo aplikacijo (bundle identifier).
Options	Slovar	Neobvezen podatek vendar kljub temu omogoče lepo vrsto dodatnih nastavitev. Možni ključi: *DisableVolumeButtons*, *DisableSleepWakeButton*, *DisableAutoLock*, *DisableTouch* in drugi.
UserEnabledOptions	Slovar	Neobvezno. Možni ključi: *VoiceOver*, *Zoom*, *InvertColors*, *AssistiveTouch*.

Tabela 4.1: Seznam ključev za nastavitev tipa *com.apple.app.lock*.

Koda 4.1: Izsek konfiguracijskega profila.

```
<array><dict>
      <key>App</key><dict>
        <key>Identifier</key>
        <string>si.cvetan.vFlyer</string>
        <key>Options</key>
        <dict>
          <key>DisableVolumeButtons</key>
          <true/>
          <key>DisableSleepWakeButton</key>
          <true/>
          <key>DisableAutoLock</key>
          <true/>
        </dict></dict>
      <key>PayloadDescription</key>
      <string>vFlyer Single APP lock.</string>
      <key>PayloadUUID</key>
      <string>DF3ED778-F94B-4D86-87E8-2177C32425F4</string>
      <key>PayloadType</key>
      <string>com.apple.app.lock</string>
      <key>PayloadDisplayName</key>
      <string>vFlyer Single APP lock</string>
      <key>PayloadVersion</key>
      <integer>1</integer>
      <key>PayloadOrganization</key>
      <string>Damjan Cvetan</string>
      <key>PayloadIdentifier</key>
      <string>si.cvetan.vflyer.lock</string>
    </dict></array>
```

4.2 Oddaljeno upravljanje

Upravljanje aplikacije, kjer je naprava v kiosk načinu in aplikacija v predstavitvenem načinu, je možno le na daljavo, pri čemer bomo pri odkrivanju ustreznih naprav v omrežju in za vzpostavitev povezave uporabili pristop konfiguracije brez konfiguracije. Uporabniku aplikacije v kontrolnem načinu bomo s tem prihranili mnogo časa in truda, ki bi ga sicer moral porabiti, če bi podatke za povezavo vnašal ročno.

4.3 Orodja in tehnologije

4.3.1 Objective-C

Objective-C je je programski jezik, ki se večinoma uporablja za pisanje programske opreme za OS X in iOS. Objective-C je nadgradnja programskega jezika C in omogoča objektno orientirano programiranje in dinamično izvajanje. Od C jezika je podedoval sintakso ter primitivne tipe in dodal sintakso za določanje razredov, metod in atributov.

Koda 4.2: Primer programa napisanega v Objective-C.

```
1  @interface Hello:NSObject
2    - (void) say;
3  @end
4
5  @implementation Hello
6  - (void) say {
7     NSLog(@"Hello, world!");
8  }
9  @end
10
11 int main(int argc, char *argv[]) {
12     @autoreleasepool {;
13         [[Hello new] say];
```

```
14      }
15      return 0;
16  }
```

4.3.2 XCode

Xcode je napredna aplikacija, ki razvijalcu programske opreme omogoča enostavno programiranje, prevajanje, razhroščevanje in poganjanje programov, ki je močno integrirana s Cocoa in Cocoa Touch frameworks.

Slika 4.1: Xcode ikona.

Xcode vključuje Xcode Integrated development environment (IDE), LLCM compiler, Instruments, iOS Simulator, zadnje izdaje OS X in iOS SDK in veliko ostalih dodatkov.

4.3.3 Bonjour

OS X in iOS ponujata več Application Programming Interface (API) plasti razvoja aplikacije z Bonjour arhitekturo. Vse plasti ponujajo možnosti za objavo, odkrivanje in razreševanje omrežnih storitev. Na sliki 4.2 so predstavljene posamezne plasti, kjer se dobro vidi, da je na spodnji plasti mDNS strežnik in programer nima potrebe, da bi neposredno delal z njim.

NSNetService in NSNetServiceBrowser

Razreda *NSNetService* in *NSNetServiceBrowser* sta del Foundation framework v Cocoa in ponujata vmesnike za objavo in odkrivanje storitev na omrežju.

NSNetService predstavlja celotno Bonjour arhitekturo, ki jo potrebujemo pri implementacji. Ta plast zadovoljuje večina Cocoa razvijalecev, vsi ostali, ki potrebujejo večji nadzor nad izvajanjem, se lahko poslužujejo DNS-SD API.

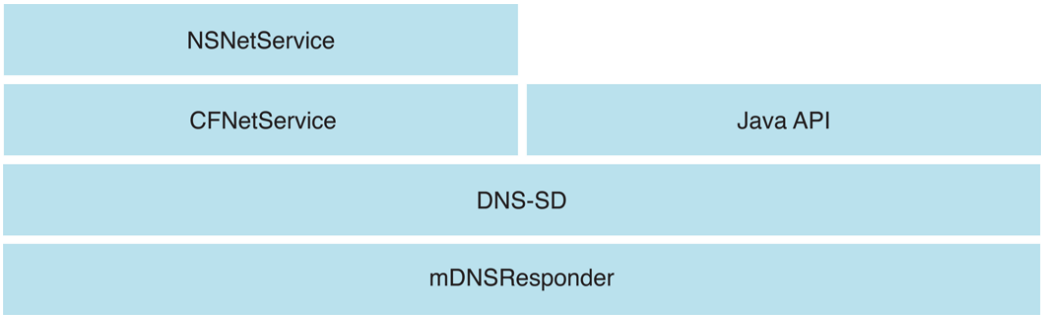
CFNetServices

CFNetServices API ponuja podobne vmesnike in funkcionalnosti kot *NSNetService* vendar s to razliko, da so napisani v Core Foundation stilu. Te vmesnike se uporablja v primeru razvoja programov Core Fundation plasti OS X in iOS.

DNS Service Discovery

V */usr/include/dns_sd.h* je definiran DNS-SD API, ki ponuja nizko nivojno komunikacijo BSD vtičnikov za Bonjour arhitekturo. Predstavlja in deluje kot vmesni člen med programom in mDNS in DNS strežnikoma.

Programiranja na tej plasti se lotimo v primerih, ko želimo pisati kodo, ki bo uporabna še na drugih sistemih, ki niso združljivi z iOS in OS X ali pa, ko želimo dostop do funkcij in parametrov, ki nam v zgornjih plasteh niso na voljo.



Slika 4.2: Plasti API za Bonjour. Vir: iOS Developer Library, Bonjour API Architecture. [10]

4.3.4 Komunikacija strežnik - odjamalec

Ko naprave vzpostavijo povezavo, imamo na voljo abstraktni razred *NSStream*, ki predstavlja vhodni in izhodni tok podatkov na obeh straneh. Način komunikacije med napravama poteka tako, da na strani pošiljatelja ustvarimo slovar iz potrebnih ključev in vrednosti, ki predstavljajo različne ukaze in informacije potrebne za upravljanje z oddaljeno napravo.

Slovar tipa *NSDictionary* pred pošiljanjem z uporabno *NSKeyedArchiver* kodiramo v neodvisni format, ki ga je moč shraniti v datoteko. Tako kodirane podatke ovijemo s kombinacijo bitov, ki oznanjajo začetek in konec kodiranih podatkov. Tako lahko na drugi strani podatkovnega toka vemo, kdaj se je prenos za aplikacijo pomembnih podatkov začel in kdaj končal.

Na drugi strani pridobljene podatke izluščimo iz ovoja in jih dekodiramo s pomočjo *NSKeyedUnarchiver* in kot rezultat dobimo *NSDictionary*, ki je bil ustvarjen na strani pošiljatelja. Dobljeni slovar nato pregledamo in glede na ključe in vrednosti ključev ustrezno reagiramo.

Koda 4.3: Koda za pripravo podatkov za pošiljanje.

```
1 - (NSData *) dataForAction:(NSString *) action andInfo:(
     NSDictionary *) info {
2   NSDictionary *dict = [[NSDictionary alloc]
     initWithObjectsAndKeys:action, @"action", info, @"info",
     nil];
3   return [NSKeyedArchiver archivedDataWithRootObject:dict];
4 }
```

4.4 Način delovanja

4.4.1 Krmilni način

Da lahko administrator ureja naprave v predstavitvenem načinu, mora vsaj na eni napravi aplikacijo uporabljati v krmilnem načinu. Ta način uporab-

niku predstavi seznam vseh naprav v omrežju, ki imajo v uporabi vFlyer aplikacijo v predstavitvenem načinu. Da lahko iščemo naprave, uporabimo razred *NSNetServiceBrowser*, ki definira vmesnik za iskanje storitev na omrežju z uporabo mDNS.

Koda 4.4: Nastavitev iskanja storitev.

```
1 NSNetServiceBrowser * serviceBrowser;
2 serviceBrowser = [[NSNetServiceBrowser alloc] init];
3 [serviceBrowser setDelegate:self];
4
5 [serviceBrowser searchForServicesOfType:kNetServiceType
     inDomain:@""];
```

Uporabnik se glede na ponujeni seznam odloči, katero napravo želi administrati. Ob izbiri željene naprave se med njima vzpostavi prosti tok podatkov v obe smeri, ki omogoča komunikacijo in izmenjavo podatkov. Administratorju se predstavi novo okno z opravili:

- Identifikacija naprave, ki na izbrani napravi prikaže identifikacijski zaslon ter s tem zakrije trenutne letake z barvo in napisom o imenu naprave in naslovu IP.

- Dodajanje novega letaka, ki omogoči izbiro letaka iz zbirke fotografij na krmilni napravi. Ob izbiri slike se le ta prenese na predstavitveno napravo in postavi na zadnje mesto v seznamu.

- Brisanje obstoječega letaka administratorju dovoljuje brisanje obstoječih letakov.

- Urejanje vrstnega reda administratorju omogoči nastavitev poljubnega vrstnega reda letakov na napravi.

4.4.2 Predstavitveni način

Za uporabnika

Predstavitveni način delovanja aplikacije na videz ne predstavlja veliko funkcionalnosti, saj aplikacija uporabniku na videz ponuja le brskanje po digitalnih letakih, ki so urejeni po straneh tako, da je vsak letak na svoji strani. Letaki so namreč grafično postavljeni na *UIScrollView*, ki ima omogočen atribut *pagingEnabled*. V takem primeru se pomik ustavi na večkratnikih dolžine vidnega območja in uporabniku da občutek premikanja po straneh.

Za administratorja

Uporabnik takšnega načina ne vidi, ker strežnik deluje v ozadju in lahko sprejema povezave drugih naprav. Strežnik smo nastavili tako, da namesti vtičnik, ki posluša za nove vhodne povezave na naključno izbranih prostih vratih, saj zaradi uporabe DNS-SD ne potrebujemo v naprej določene številke vrat.

Ko enkrat imamo delujoč strežnik je potrebno poskrbeti, da to storitev oglašujemo oziroma objavimo v našem omrežju. To naredimo s pomočjo API za Bonjour in sicer s pomočjo *NSNetService* razreda.

Koda 4.5: Ustvarjanje vtičnika

```
1 listeningSocket = CFSocketCreateWithNative(NULL,
2   fd4,
3   kCFSocketAcceptCallBack,
4   ListeningSocketCallBack,
5 &context);
```

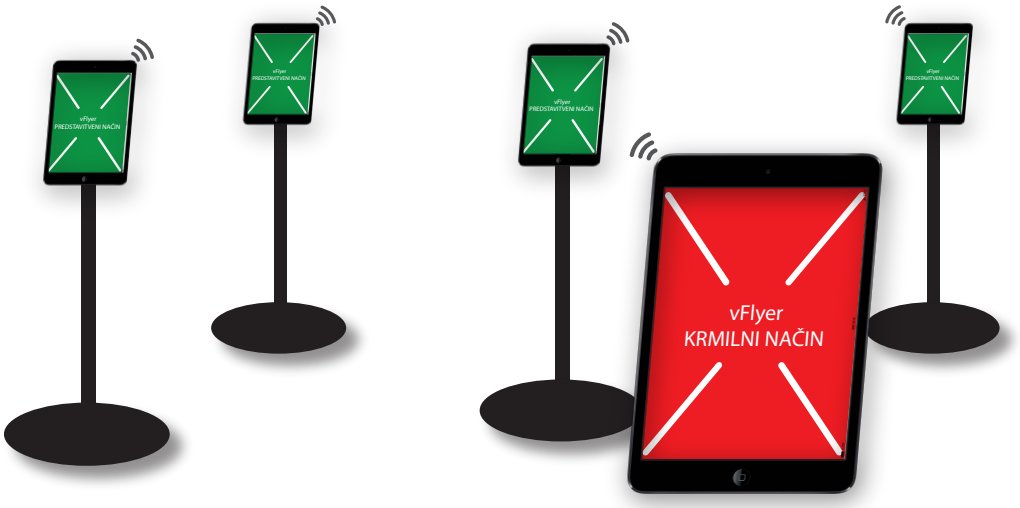
Koda 4.6: Bonjour nastavitev.

```
1 NSNetService * netService;
2
3 netService = [[NSNetService alloc] initWithDomain:@"" type:
     kNetServiceType name:[[NSUserDefaults standardUserDefaults]
```

```
        stringForKey:kSettingKeyDeviceName] port:ntohs(sin.
       sin_port)];
4
5  [netService scheduleInRunLoop:[NSRunLoop currentRunLoop]
       forMode:NSRunLoopCommonModes];
6
7  [netService setDelegate:self];
8  [netService publish];
```

4.5 Shema namestitve

Da izkoristimo rešitev do popolnosti, potrebujemo v omrežju dve ali več naprav z aplikacjo v izvajanju. Na eni napravi nastavimo delovanje aplikacije v predstavitveni način ter na drugi v krmilni način. V primeru večjega števila naprav je dovolj, da imamo le eno napravo z aplikacijo v krmilnem načinu.



Slika 4.3: Namestitev v okolju z večjim številom naprav. Na štirih napravah deluje aplikacija v predstavitvenem načinu, na eni napravi pa v krmilnem načinu.

Na sliki 4.3 naprava v krmilnem načinu dinamično posodablja seznam naprav, ki so v omrežju na voljo za upravljanje. Primerne so vse naprave, ki ustrezno odgovorijo na poziv s strani krmilne aplikacije, ki poda zahtevo po ustrezni PTR vrednosti.

S takšnim pristopom ni potrebe po poznavanju naslova IP, saj moramo poznati le ime oddaljene naprave, katerega lahko v nastavitvah (poglavje 5.5) prilagodimo svojim željam.

Lahko se nam zgodi, da imamo v omrežju dve napravi v krmilnem načinu. To ne predstavlja nobene težave, saj se vsa vsebina na oddaljeni napravi vedno osveži s krmilno napravo v primeru uspešno vzpostavljene povezave. Možno je urejati več oddaljenih naprav z večjim številom krmilnih naprav. Pri tem tudi ni potrebe, da bi vso gradivo hranili le na eni krmilni napravi.

Poglavje 5

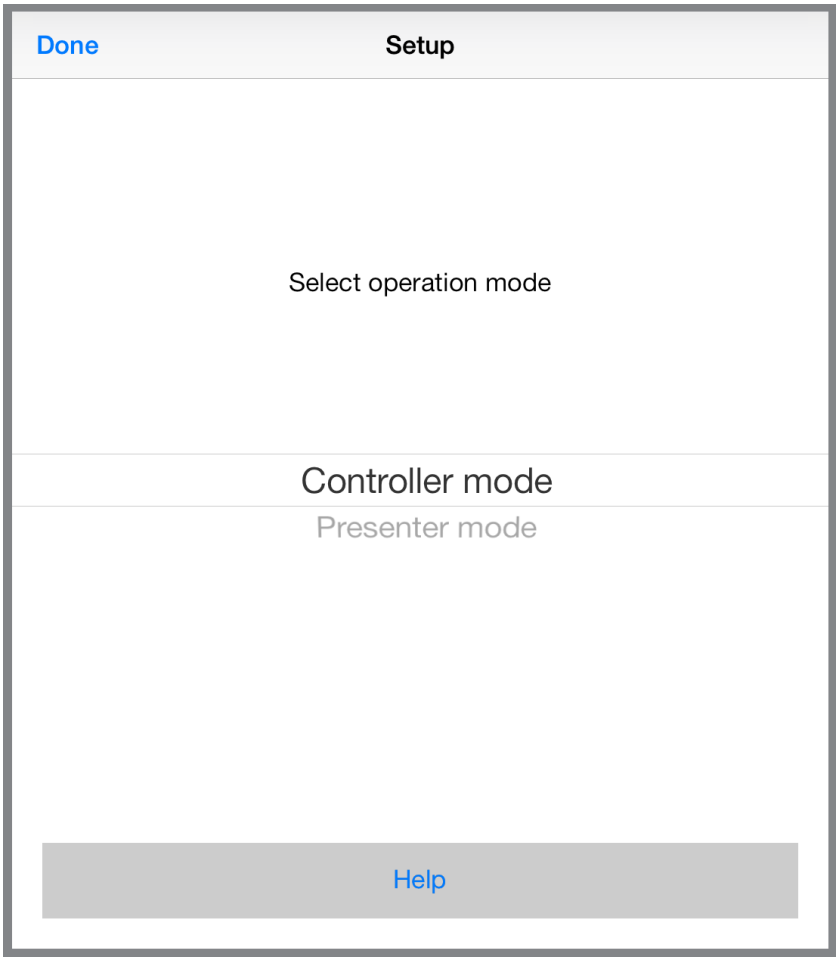
Način uporabe aplikacije

5.1 Namestitev in prva uporaba

Slika 5.1: Quick Response Code (QR Code) s povezavo `http://vflyer.cvetan.si/` do namestitve.

Aplikacijo si lahko na svojo napravo namestite z uporabo QR Code iz slike 5.2 katero preberete s pomočjo ustreznega bralnika na napravi ali pa neposredno obiščete naslov `http://vflyer.cvetan.si/`.

Po uspešni namestitvi lahko aplikacijo poženete in ob prvem zagonu vas bo pričakal začetni zaslon za izbiro načina delovanja. Način delovanja lahko kasneje spremenite v globalnih nastavitvah (poglavje 5.5).



Slika 5.2: Prvi zagon aplikacije.

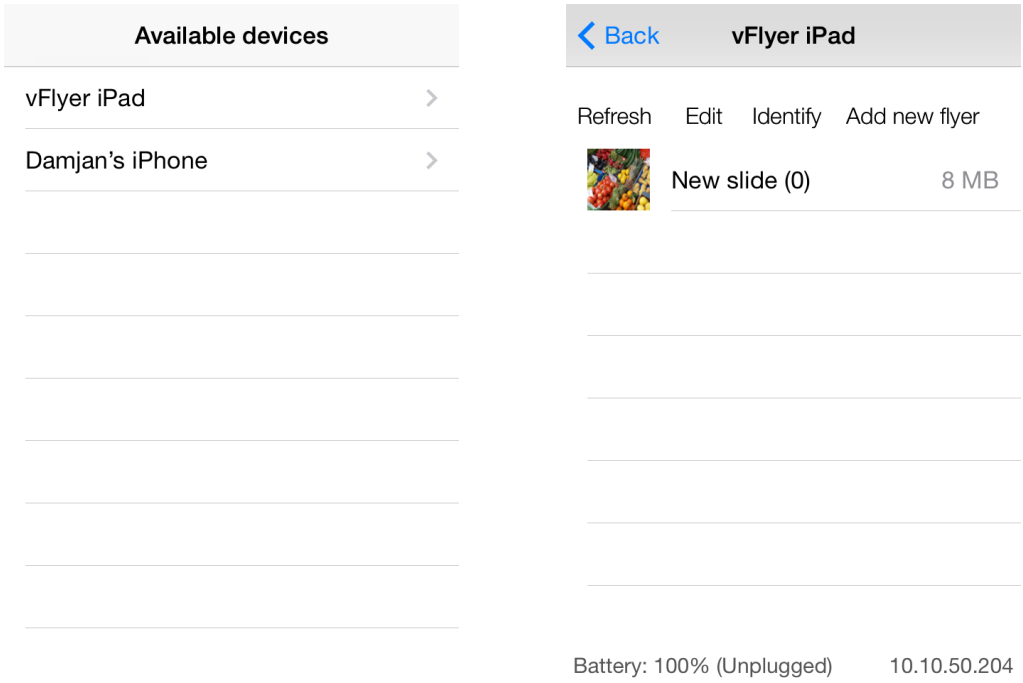
5.2 Priprava vsebine

Vsebino oziroma letake lahko predhodno pripravite na osebnem računalniku ali katerem koli drugem okolju s katerim lahko ustvarite sliko, ki predstavlja vaš letak. Letak nato prenesete na napravo v aplikacijo *Photos*. Prenos lahko opravite ali preko programa iTunes ali pa tako, da uporabite enega izmed mnogih načinov prenosa slike preko spleta. Uporabite lahko e-pošto, spletni ali FTP strežnik, Google Drive, DropBox ali kaj podobnega.

5.3 Upravljanje z vsebino

Z vsebino znotraj aplikacij, ki delujejo v predstavitvenem načinu, upravljate z isto aplikacijo na drugi napravi, ki jo uporabljate v krmilnem načinu.

Aplikacija vam na začetku ponudi seznam naprav, ki so na voljo v omrežju in delujejo v predstavitvenem načinu. Seznam se posodablja avtomatično glede na prihajanje in odhajanje naprav na omrežju. S tem je seznam vedno osvežen in vsebuje le naprave, ki so trenutno na voljo.



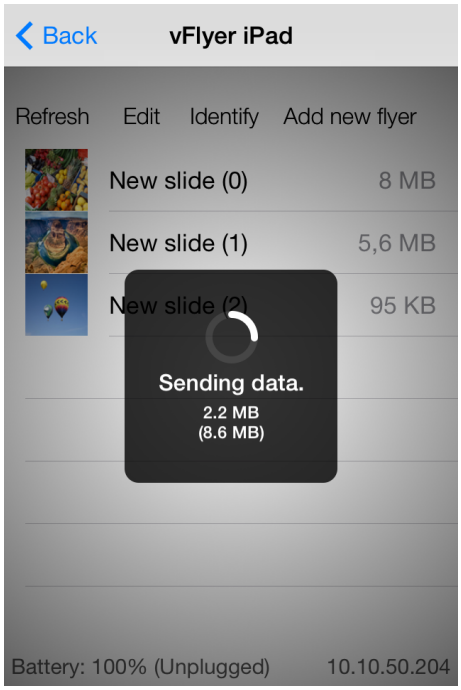
(a) Seznam ustreznih naprav v omrežju. (b) Upravljanje z vsebino.

Slika 5.3: Izbira naprave.

Ob izbiri naprave se med obema napravama vzpostavi komunikacija in odpre se vam zaslon za urejanje vsebine na izbrani napravi. Zaslon vam omogoča:

- Osvežitev vsebine, ki ponovno pridobi podatke o vsebini s strani oddaljene naprave.

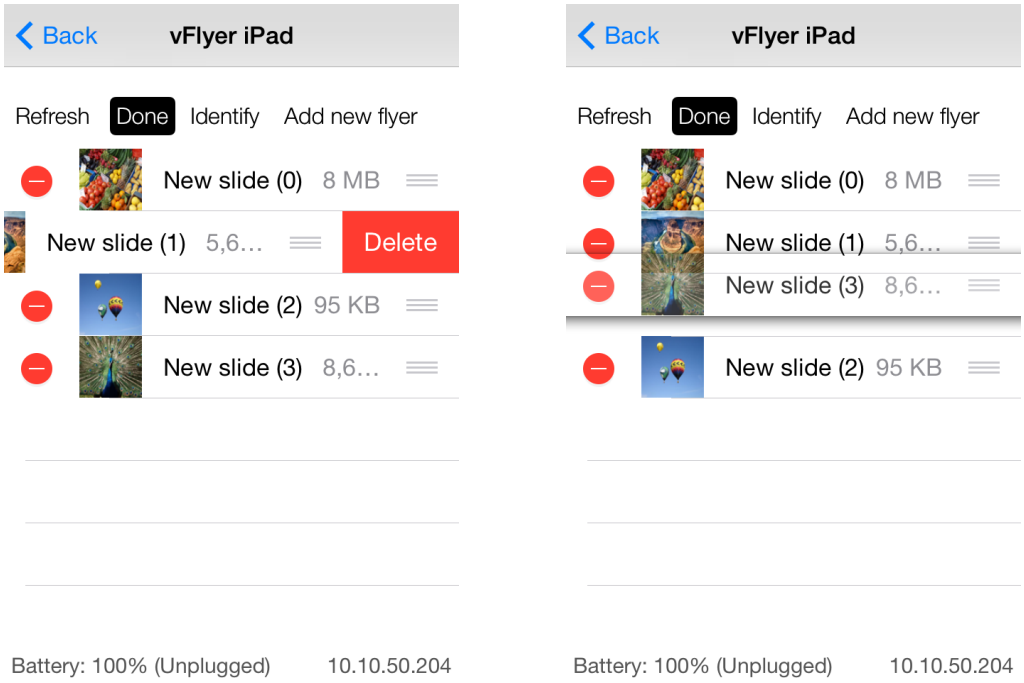
- Urejanje vsebine, ki vam omogoča brisanje in urejanje vrstnega reda letakov.
- Identificiranje, ki povzroči, da se na oddaljeni napravi prikaže njeno ime in naslov IP.
- Dodajanje novega letaka, ki novi letak prenese na oddaljeno napravo in ga doda na konec seznama.
- Prikaz imena (zgoraj), stanje baterije (levo spodaj) in naslov IP (desno spodaj) oddaljene naprave.



Slika 5.4: Pošiljanje nove vsebine na oddaljeno napravo.

5.4 Predstavitev vsebine

V primeru delovanja aplikacije v predstavitvenem načinu se na napravi prikažejo letaki, med katerimi je mogoče prehajati z vlečenjem prsta po ekranu levo ali



(a) Izbris letaka.

(b) Spreminjanje vrstnega reda.

Slika 5.5: Brisanje letaka in spreminjanje vrstnega reda.

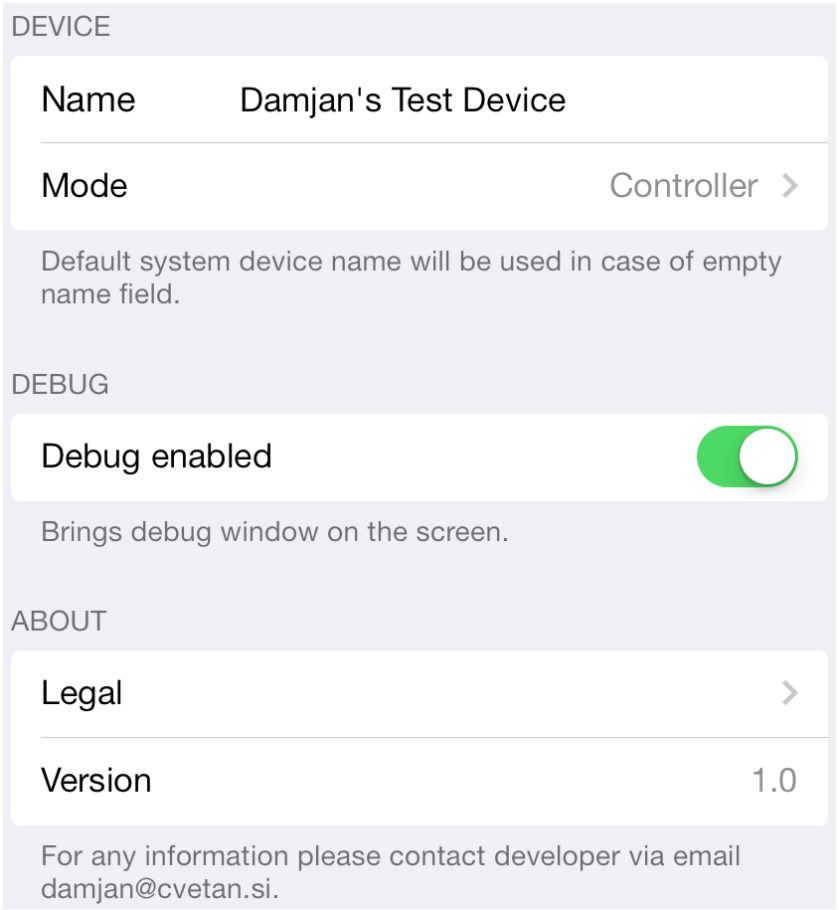
desno. Takšen gib nam namreč odkrije ostale letake, ki so postavljeni levo ali desno od trenutno aktivnega, kot to prikazuje slika 5.6. Za enkrat ni omejitve koliko letakov lahko dodamo na napravo, je pa to seveda pogojeno s prostim prostorom na napravi in velikostjo posameznega letaka.

Slika 5.6: Postavitev letakov na trak.

5.5 Nastavitve

Nastavitve aplikacije so na voljo v globalnih nastavitvah naprave. Do nastavitev pridemo tako, da odpremo nastavitve naprave in iz seznama aplikacij izberemo aplikacijo vFlyer. Nastavljamo lahko:

- Ime naprave, ki je na začetku enako sistemskemu imenu naprave.
- Način delovanja naprave (kontrolni, predstavitveni).
- Prikaz dnevniških zapisov na zaslonu, ki omogočajo lažje odpravljanje težav.



Slika 5.7: Nastavitve aplikacije.

Poglavje 6

Sklepne ugotovitve

Diplomo smo zasnovali kot reševanje problema upravljanja aplikacije, ki deluje na napravi v kiosk načinu. Želeli smo poiskati praktičen, učinkovit in uporabniku prijazen način. Zastavili smo zasnovo rešitve z oddaljenim povezovanjem, pri čemer ne poznamo niti naslova ponora niti številke vrat na katerih deluje storitev za povezovanje. Cilj smo dosegli z uporabo konfiguracije brez konfiguracije, ki kot spoznamo skozi delo, deluje s pomočjo različnih storitev (naslavljanje, mDNS, DNS-SD) in njihovih lastnosti.

Praktična uporabnost takšne arhitekture je velika in je v pomoč predvsem končnemu uporabniku, ki mu v primeru pravilne izvedbe močno izboljša uporabniško izkušnjo. Uporabnik se mora na koncu le odločiti, katero napravo iz seznama bo uporabil za tiskanje dokumenta ali s katero osebo na sestanku bo delil slikovno gradivo.

Skozi delo smo spoznali in praktično prikazali uporabnost konfiguracije brez konfiguracije. Sedaj na probleme povezovanja naprav okoli nas gledamo drugače in se sprašujemo, zakaj ta tehnologija ni bolj razširjena. Pričakovati bi namreč bilo, da bi že v preteklem koledarskem letu lahko pozabili na naslove in številke vrat, ki jih potrebujemo za povezavo na oddaljen računalnik. Podjetje Apple je s svojim operacijskim sistemom OS X in iOS na dobri poti, saj tehnologijo srečamo na vsakem koraku uporabnikov teh sistemov. Konfiguracijo brez konfiguracije uporablja predvajalnik glasbe iTunes za de-

ljenje glasbe med napravami, tehnologija AirPlay za odkrivanje naprav, ki omogočajo deljenje zaslona in še bi lahko kaj našli. Uporabniki ostalih bolj ali manj razširjenih operacijskih sistemov se s to arhitekturo v večini primerov srečajo le pri uporabi tiskalnikov.

Menim, da bi morali konfiguracijo brez konfiguracije vgrajevati v vse naprave in programe, ki od nas zahtevajo oziroma omogočajo oddaljeno povezovanje.

Literatura

[1] Avahi. Dostopno na:
http://avahi.org/, 2014. (19).

[2] Avahi (software). Dostopno na:
http://en.wikipedia.org/wiki/Avahi_(software), 2014. (19).

[3] Internet assigned numbers authority. Dostopno na:
http://www.iana.org/, 2014. (7).

[4] iOS. Dostopno na:
http://en.wikipedia.org/wiki/IOS, 2014. (3).

[5] Kiosk software. Dostopno na:
http://en.wikipedia.org/wiki/Kiosk_software, 2014. (2).

[6] L. Esibov A. Gulbrandsen, P. Vixie. A DNS RR for specifying the location of services (IETF RFC 2782). Dostopno na:
http://tools.ietf.org/html/rfc2782/, Februar 2000. (6).

[7] IANA. Special-use IPv4 addresses (IETF RFC 3330). Dostopno na:
http://tools.ietf.org/html/rfc3330/, Sptember 2002. (11).

[8] Apple Inc. Bonjour for developers. Dostopno na:
https://developer.apple.com/bonjour/, 2014. (12, 20).

[9] Apple Inc. Configuration profile key reference. Dostopno na:
https://developer.apple.com/library/ios/featuredarticles/

iPhoneConfigurationProfileRef/Introduction/Introduction.html, 2014. (23).

[10] Apple Inc. ios developer library, bonjour api architecture. Dostopno na: https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/NetServices/Articles/programming.html, 2014. (28).

[11] Apple Inc. ios developer library, bonjour operations. Dostopno na: https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/NetServices/Articles/NetServicesArchitecture.html, 2014. (16, 18).

[12] L. Eggert M. Cotton, J. Touch, M. Westerlund, and S. Cheshire. Internet assigned numbers authority (IANA) procedures for the management of the service name and transport protocol port number registry (IETF RFC 6335). Dostopno na: http://tools.ietf.org/html/rfc6335/, Avgust 2011. (7).

[13] P. Mockapetris. Domain names – implementation and specification (IETF RFC 1035). Dostopno na: http://tools.ietf.org/html/rfc1035/, November 1987. (6, 7, 7).

[14] B. Aboba S. Cheshire and E. Guttman. Dynamic configuration of IPv4 link-local addresses (IETF RFC 3927). Dostopno na: http://tools.ietf.org/html/rfc3927/, Maj 2005. (12).

[15] D. Steinberg S. Cheshire. *Zero Configuration Networking: The Definitive Guide*. Sebastopol, O'Reilly, 2006. (9).

[16] M. Krochmal S. Cheshire. Multicast (DNS) (IETF RFC 6762). Dostopno na: http://tools.ietf.org/html/rfc6762/, Februar 2013. (13, 14).

[17] M. Krochmal S. Cheshire. DNS-based service discovery (IETF RFC 6763). Dostopno na: `http://tools.ietf.org/html/rfc6763/`, Februar 2013. (14).

[18] Cheshire Stuart. Zero configuration networking. Dostopno na: `http://www.zeroconf.org/`, 2014. (3, 9).

[19] Lennart Poettering Trent Lloyd. Using avahi the "right way"(prosojnice, linux conference australia), Januar 2007. (19).